MEMCOPY

Carefully manage size of destination buffer

Sean Barnum, Cigital, Inc. [vita[1]]

Copyright © 2007 Cigital, Inc.

2007-03-26

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 4269 bytes

Attack Category	• Malicious Input
Vulnerability Category	• Buffer Overflow
Software Context	• Memory Management
Location	• string.h • CHtmlStream (MFC)
Description	When memory contents are being copied from one location to another, a buffer overflow will occur if the destination buffer is not large enough to hold the amount of data being copied. When the destination buffer is statically allocated, static analysis should reveal whether a buffer overflow can occur.
APIs	**Function Name** / **Comments** _mbsnbcpy bcopy — bytes, not strings CHtmlStream::Memcpy — lpMemTarget, lpMemSource CMemFile::Memcpy — lpMemTarget, lpMemSource CopyMemory — Destination, Source memcpy — bytes, not strings MoveMemory — allows overlapping memory blocks
Method of Attack	An attacker that can control the number of bytes being copied to the destination buffer may be able to cause a buffer overflow, which may crash the application or overwrite other important data.
Exception Criteria	When the destiation buffer is statically allocated and the number of bytes being copied is less than the size of the buffer.

1. http://buildsecurityin.us-cert.gov/bsi-rules/35-BSI.html (Barnum, Sean)

<table>
<tr><td>Solutions</td><td>
<table>
<tr><th>Solution Applicability</th><th>Solution Description</th><th>Solution Efficacy</th></tr>
<tr><td>Applicable whenever memory blocks are being copied.</td><td>Ensure that the number of bytes being copied into a buffer is at most the size of the buffer.</td><td>Effective.</td></tr>
</table>
</td></tr>
<tr><td>Signature Details</td><td>unsigned char * _mbsnbcpy(unsigned char * strDest, const unsigned char * strSource, size_t count)
void bcopy(const void *src, void *dst, int n)
virtual BYTE* Memcpy(BYTE* lpMemTarget, const BYTE* lpMemSource, UINT nBytes)
void CopyMemory(PVOID Destination, const VOID* Source, SIZE_T Length)
void *memcpy(void *s1, const void *s2, size_t n)
void MoveMemory(PVOID Destination, const VOID* Source, SIZE_T Length)</td></tr>
<tr><td>Examples of Incorrect Code</td><td><pre>[...]
char *dest, *src;
[...]
memcpy(dest, src, strlen(src));
[...]</pre></td></tr>
<tr><td>Examples of Corrected Code</td><td><pre>[...]
char *dest, *src;
int dest_sz, src_sz;
[...]
memcpy(dest, src, (dest_sz >=
src_sz)?src_sz:dest_sz);
[...]</pre></td></tr>
<tr><td>Source References</td><td>• http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnsecure/html/appsec.asp[2]</td></tr>
<tr><td>Recommended Resources</td><td></td></tr>
<tr><td>Discriminant Set</td><td>
<table>
<tr><td>Operating System</td><td>• Any</td></tr>
<tr><td>Language</td><td>• C++</td></tr>
</table>
</td></tr>
</table>

Cigital, Inc. Copyright

Copyright © Cigital, Inc. 2005-2007. Cigital retains copyrights to this material.

Permission to reproduce this document and to prepare derivative works from this document for internal use is granted, provided the copyright and “No Warranty” statements are included with all reproductions and derivative works.

For information regarding external or commercial use of copyrighted materials owned by Cigital, including information about “Fair Use,” contact Cigital at copyright@cigital.com[1].

1. mailto:copyright@cigital.com

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.